

PRV

PATENT- OCH REGISTRERINGSVERKET
Patentavdelningen

PCT/SE 2004/000103

**Intyg
Certificate**

REC'D 10 FEB 2004

WIPO

PCT

Härmed intygas att bifogade kopior överensstämmer med de handlingar som ursprungligen ingivits till Patent- och registreringsverket i nedannämnda ansökan.

This is to certify that the annexed is a true copy of the documents as originally filed with the Patent- and Registration Office in connection with the following patent application.



(71) Sökande *Xelerated AB, Stockholm SE*
Applicant (s)

(21) Patentansökningsnummer *0300198-9*
Patent application number

(86) Ingivningsdatum *2003-01-28*
Date of filing

Stockholm, 2004-02-05

*För Patent- och registreringsverket
For the Patent- and Registration Office*

Marita Öun

Marita Öun

Avgift
Fee

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH RULE 17.1(a) OR (b)

**PATENT OCH
REGISTRERINGSVERKET
SWEDEN**

Postadress/Address
Box 5055
S-102 42 STOCKHOLM

Telefon/Phone
+46 8 782 25 00
Vx 08-782 25 00

Telex
17978
PATOREG S

Telefax
+46 8 666 02 86
08-666 02 86

A METHOD IN PIPELINED DATA PROCESSING

TECHNICAL FIELD

5 The invention concerns a method in a processor, in which data is processed in a pipelined manner, the data being included in a plurality of contexts, comprising a first context, in addition to which a plurality of operations is adapted to be executed on the contexts.

10 BACKGROUND

In data processing the rate of the data process is an important factor for an efficient processor. A way to allow a high rate of data passing through a processor is to perform pipelined processing of the data, i.e. allowing the process to be executed on
15 one set of data before the process of previous sets of data are finalized. Such processes are typically carried out in a number of stages at which operations are executed on contexts including sets of data.

SUMMARY OF THE INVENTION

20

It is an object of the present invention to increase the rate of data being processed in a data processor.

25

This object is reached by a method of the type mentioned initially, characterized in that the method comprises commencing an execution on the first context of a second operation before a previously commenced execution on the first context of a first operation is completed. This means that in addition to pipelined processing of the data stream, the execution of each operation is pipelined. Thus, a form of "two dimensional" pipelining is achieved. On one hand the data stream is pipelined, and on
30 the other hand, the "operation stream" is pipelined.

The object is also reached with a method of the type described initially, including executing an initial operation step of a first operation on the first context, and subsequently commencing an execution on the first context of an initial operation step of a second operation before an execution on the first context of a following operation step of the first operation is completed. Preferably, each context passes a plurality of consecutive stages, whereby the initial operation step of the first operation is executed on the first context at a first stage, the following operation step of the first operation is executed on the first context at a second stage, and the initial operation step of the second operation is executed on the first context at the second stage.

Thus, the operation pipelining can be done in a number of steps in the pipelined data processor. If each operation is performed in N steps, the processor can run in a higher frequency, i.e. N times higher than in a case with no operation pipelining. This means that during a certain time interval more instructions can be executed. This can provide for a higher data bandwidth throughput in the data stream pipeline.

Preferably, the method comprises receiving at the second stage a result of an execution of the initial operation step of the first operation. This provides for commencing the execution of an operation at one stage, and continuing the execution at a consecutive stage.

Preferably, where the first operation comprises a partial operation of executing an instruction and a partial operation of writing a result of the said instruction execution into a destination in a register, and the second operation comprises the partial operation of fetching an operand, the method can comprise the following steps: Determining if a position in the register, from which the operand is to be fetched in the second operation, is identical with the destination of the partial operation, of the first operation, of writing a result. If the result of the step of determining is negative,

fetching the operand from the register. If the result of the step of determining is positive, fetching the result of the said instruction execution. Thereby it is possible to initiate an operation before an operation initiated previously on the same context, without having to wait for the previously initiated operation to be completed. This will facilitate increasing the data processing rate of the processor.

BRIEF DESCRIPTION OF FIGURES

Below, the invention will be described in detail with reference to the drawings, in which

- fig. 1, 3, 4 and 6 show block diagrams of processes in pipelined data processors,
- fig. 2 shows a table correlating objects in the process of fig. 1, and
- fig. 5 shows a schematic example of a program code for functions depicted in fig. 4.

DETAILED DESCRIPTION

Fig. 1 depicts schematically a data processing pipeline 1, in a data processor. The pipeline comprises a plurality of stages 2a-2f. For the purpose of this presentation only six stages are shown, whereas any number of stages could be used in the pipeline, and in reality a considerably larger number of stages would be used.

At each clock cycle of the data processor, a context 3, including data to be processed, is received at each stage from the preceding stage, i.e. the stage immediately to the left in fig. 1. In fig. 1 each stage 2a-2f is depicted at a time corresponding to one clock cycle after the preceding stage, which is indicated by the time axis T. Thus, fig. 1 shows any stage presented at a time occurring after the time of presentation of the preceding stage, the difference in time corresponding to the duration of one clock cycle, Δt . Hence, the context 3 in any stage in fig. 1 corresponds to the

context in any other stage in fig. 1. Of course, at the same point in time the stages have different contexts, which is a feature that makes the data processing pipelined.

Each context 3 includes a packet, with data, which may be processed by the processor, a register file, flags, and an instruction counter or instruction pointer, e.g. a row instruction pointer, as described in the international patent applications PCT/SE01/01134 and PCT/SE01/01133, each with the applicant of this application and included herein by reference.

10 In the data processing pipeline a number of operations are performed in connection to each context. Each operation consists of a number of partial operations. In a plurality of stages 2a-2e, during each clock cycle, partial operations are performed in connection to the context 3. Each operation can comprise executing in a logic unit 5a-5e in the respective stage an instruction stored in an instruction memory, not
15 shown in fig. 1. The logic unit could comprise an ALU.

Each operation comprises a number of steps, each comprising one or more of the partial operations. In pipelined data processing, an operation typically comprises the partial operations: (i) instruction fetch, (ii) instruction decoding, (iii) operand fetch,
20 (iv) execution, (v) branch, and (vi) write back. These partial operations could be allocated to any number of operation steps. Suitably, an operation could contain two steps, the first containing partial operations (i) to (iii) and the second containing partial operations (iv) to (vi).

25 For a better understanding of the concept of the invention, each operation in this example comprises three operation steps, for this presentation referred to as an initial operation step 6a, 7a, 8a, an intermediate operation step 6b, 7b, 8b, and a final operation step 6c, 7c, 8c. The intermediate and the final operations step are also referred to as "following operation step".
30

In general, each operation can comprise any number of operation steps. It should also be kept in mind that the context can alternatively be received by a stage of the processing pipeline without any partial operations being performed on it, or on parts of it.

5

A first context 3, which could be any context in the processor, is received at a first stage 2a, which could be any stage in the processor. In the first stage 2a, a first initial operation step 6a of a first operation is performed on the first context 3. A first initial operation step result R6a is generated as a result of the first initial operation step 6a being performed on the first context 3.

10

Subsequently, in a second stage 2b, the first context 3, modified by the first initial operation step 6a, is received from the first stage 2a. The modified first context 3 comprises the first initial operation step result R6a. It should be noted that the pipeline is adapted so that when a context is received in a stage from a previous stage, the previous stage receives another context, as described in the above referenced international patent application PCT/SE01/01134.

15

In a pipelined manner, essentially simultaneously with the first context 3 being received in the second stage 2b, a second context, not shown, is received at the first stage 2a.

20

In the second stage 2b, a first intermediate operation step 6b of the first operation is performed on the first context 3, based on the first initial operation step result R6a. As a result a first intermediate operation step result R6b is generated.

25

During the same clock cycle, at $t+\Delta t$, the initial operation step 6a of the first operation is executed on the second context. Thus, the initial operation step 6a of the first operation is executed on the second context before the execution on the first context 3 of the following operation step 6b of the first operation is completed. In other

30

words, an execution on the second context of a first operation is commenced before a previously commenced execution on the first context of the first operation is completed.

5 Also, in the second stage 2b, a second initial operation step 7a of a second operation is performed on the first context 3, and a second initial operation step result R7a is generated as a result thereof.

10 Subsequently, in a third stage 2c, the modified first context 3 is received from the second stage 2b. Thereby, the first context 3 comprises the second initial operation step result R7a and the first intermediate operation step result R6b.

In the third stage 2c, a first final operation step 6c of the first operation is performed on the first context 3, based on the first intermediate operation step result R6b.

15 Since, in this example, each operation consists of three operation steps, by the first final operation step 6c, the partial operations of the first operation on the first context 3 are completed.

20 Also, in the third stage 2c, a second intermediate operation step 7b of the second operation is performed on the first context 3, based on the second initial operation step result R7a. A second intermediate operation step result R7b is generated as a result thereof.

25 Also, in the third stage 2c, a third initial operation step 8a of a third operation is performed on the first context 3, and a third initial operation step result R8a is generated as a result thereof.

30 Fig. 2 shows a table in which the first two columns correlate operation steps and stages in the example in fig. 1. It can easily be understood that, since different steps of each operation are carried out in separate stages of the processor pipeline, a per-

son programming the instruction memory will be faced with a task that can seem complicated in cases where there are a lot of stages in the pipeline. Therefore, the processor is arranged so that all steps, 6a-6c, 7a-7c, etc, of each operation are presented to a programmer as being carried out in the same stage, 2a, 2b, etc, of the pipeline, see the third column in the table in fig. 2. This will facilitate the job of the programmer since it keeps the programming of the instruction memory clear and well-arranged. Thus, the true correlation of operation steps and stages of the processing pipeline will not be visible to the programmer.

Referring again to fig. 1, in this example, the third stage is the last stage at which an operation is initiated. A fourth, fifth and sixth stage 2d, 2e, 2f are located at the end of the pipeline. Since all steps of the second and third operation appears to a programmer of the processor to be executed in the second and third stage 2b, 2c, respectively, the stages following the third stage 2c will be invisible to the programmer.

In the fourth stage 2d, the modified first context 3 is received from the third stage 2c, whereby it comprises the third initial operation step result R8a and the second intermediate operation step result R7b. A second final operation step 7c is performed, based on the second intermediate operation step result R7b. Thereby, the partial operations of the second operation are completed. A third intermediate operation step 8b is performed, based on the third initial operation step result R8a, resulting in a third intermediate operation step result R8b.

Subsequently, in the fifth stage 2e, the modified first context 3 is received from the fourth stage 2d, whereby it comprises the third intermediate operation step result R8b, based on which a ~~third final operation step~~ 8c is performed. Thereby, the partial operations of the third operation are completed. In the sixth stage 2f the context 3 is received after completion of partial operations of three operations.

Usually, in a data processing pipeline the execution of an operation is dependent upon the result of a previous execution of another operation. According to a preferred embodiment of the invention, multiple branch executions of operations or operation steps are performed to facilitate commencing execution of subsequent operations before previously initiated operations have been completed.

For an example of multiple branch execution, fig. 3 depicts schematically a data processing pipeline 1, similar to the pipeline in fig. 1. For this presentation the pipeline comprises only five stages 2a-2e.

As in the example presented with reference to fig. 1, partial operations including the execution of instructions stored in an instruction memory, not shown in fig. 3, are performed on a context 3 by logic units 5a-5d in the stages. As in the example above, each operation comprises three operation steps: an initial operation step 6a, 7a, an intermediate operation step 6b, 7b, and a final operation step 6c, 7c. In this example, only two operations are executed.

A first context 3 is received at a first stage 2a, where a first initial operation step 6a of a first operation is performed. A first initial operation step result R6a is generated as a result of the first initial operation step 6a being performed on the first context 3.

Subsequently, in a second stage 2b, the first context 3, comprising the first initial operation step result R6a, is received from the first stage 2a. In the second stage 2b, a first intermediate operation step 6b of the first operation is performed on the first context 3, based on the first initial operation step result R6a. As a result a first intermediate operation step result R6b is generated.

In this example, the execution of a second operation is dependent upon the final result of the first operation. We assume that there are two execution paths of the second operation, both of which are initiated in a second initial operation step 7a of the

second operation. Since the second initial operation step 7a is carried out in stage 2b, before a first final operation step of the first operation has been executed, both execution paths of the second initial operation step 7a are carried out, resulting in two alternative second initial operation step results, R7a1, R7a2.

5

In a real utilization of the invention more than two execution paths are possible in an operation, whereby all paths may have to be executed or at least initiated before a subsequent operation is initiated.

10

Subsequently, in a third stage 2c, the modified first context 3 is received from the second stage 2b. Thereby, the first context 3 comprises the two alternative second initial operation step results, R7a1, R7a2, and the first intermediate operation step result R6b.

15

In the third stage 2c, a first final operation step 6c of the first operation is performed on the first context 3, based on the first intermediate operation step result R6b, whereby the partial operations of the first operation on the first context 3 are completed. Thereby, a first operation result, R6, is generated.

20

Also, in the third stage 2c, two second intermediate operation steps 7b of the second operation are performed on the first context 3, each based on one of the two alternative second initial operation step results, R7a1, R7a2. One second intermediate operation step result, R7b1, is generated as a result of the second intermediate operation steps 7b being performed on the basis of one of the two alternative second initial operation step results, R7a1. Another second intermediate operation step result, R7b2, is generated as a result of the second intermediate operation steps 7b being performed on the basis of the other of the two alternative second initial operation step results, R7a2.

25

In a fourth stage 2d, the modified first context 3 is received from the third stage 2c, whereby it comprises the first operation result, R6, and both second intermediate operation step results R7b1, R7b2. Based on the first operation result, R6, it is determined whether a second final operation step 7c should be carried out based on one or the other of the second intermediate operation step results R7b1, R7b2. When this is determined, the second final operation step 7c is performed, based on whichever of the two second intermediate operation step results R7b1, R7b2, that was determined to form a base of the second final operation step 7c. Thereby, the partial operations of the second operation are completed.

10

A number of alternatives of multiple branch execution are possible. For example, different numbers of execution paths could be performed at different steps of the same operation. Referring to the example in fig. 3, a plurality of execution paths could be performed based on each initial operation step result R7a1, R7a2, in the intermediate operation step 7b, resulting in more than two intermediate operation step results.

15

Alternatively, only one execution path could be performed in the initial operation step 7a, upon which two or more execution paths of the following, or intermediate, operation step 7b are performed.

20

In general, according to a preferred embodiment of the invention, where at least one of the operation steps of an operation comprises at least two alternative execution paths, at least two of the alternative execution paths of the operation step can be executed at a stage of the processing pipeline. Thereby, results are obtained of at least two of the executions of the alternative execution paths. Based on a result of an execution of another operation initiated before the initiation of the said operation, it is determined which one of the results of the executions of the alternative execution paths, an execution of an operation step, following said operation step comprising at least two alternative execution paths, is to be based on.

25

30

Multiple branch execution, as described above allows for the execution of an operation to commence, in spite of this execution being dependent on the result of a previously commenced execution of another operation, and the latter execution not being finalized.

For a further example of multiple branch execution we refer to fig. 4 and 5. For simplicity the example contains a data processing pipeline with only four stages as depicted in fig. 4. Fig. 5 shows a program code for the pipeline depicted in fig. 4.

In a first stage 2a a context 3 is received. In this stage 2a two partial operations of a first operation are performed regarding the context 3. One of these partial operations is a fetch partial operation 6F, including fetching an instruction in an instruction memory, (not shown). The other partial operation is a decode partial operation 6D, including decoding of the fetched operation. In fig. 5 these partial operations are executed according to rows 1 and 2.

In a second stage 2b, an execute partial operation 6E of the first operation is performed on data related to the context 3 according to the instruction fetched and decoded in the first stage 2a. Also, in the second stage 2b a second operation is commenced regarding the context 3.

The instruction executed in the execute partial operation 6E is a conditional jump, the jump depending on a value of a parameter x . In fig. 5 it can be seen on row 3 that if $x=0$, the program is continued on row L. However, the result of the execute partial operation 6E will not be known until the end of the clock cycle $t+\Delta t$, (see fig. 4), during which the context is in the second stage 2b. Therefore, referring to fig. 5, when the second operation is commenced it will not be known if the program will be on row L+1 or row 4, since it will not be known whether the execution of the instruction regarding the previously commenced operation will cause the program to

jump or not. Therefore, a multiple branch execution of the second operation is performed, involving two fetch partial operations 7F1, 7F2 of the second operation. Referring to fig. 5, one of these fetch partial operations is performed according to row 4 and the other is performed according to row L+1 in the program. Similarly, two
 5 decode partial operations 7D1, 7D2 are performed, one according to row 5 and the other according to row L+2 in the program. In a third stage 2c of the data processing pipeline, the context 3 is received and a store partial operation 6S of the first operation is performed, which, depending on whether or not a jump was performed in the preceding stage 2b, is performed according to row 6 or row L+3 in the program,
 10 (see fig. 5).

Since the result of the execute partial operation 6E is known, it can be determined which one of the instructions fetched and decoded in the second stage 2b should be executed in the third stage 2c. If no jump was made as a result of the execute partial
 15 operation 6E in the second stage 2b, the instruction fetched and decoded according to program rows 4 and 5 will be used in an execute partial operation 7E in the third stage 2c according to row 7 in the program. If a jump was made as a result of the execute partial operation 6E in the second stage 2b, an execute partial operation 7E will be performed using the instruction fetched and decoded in the second stage 2b according to program rows L+1 and L+2, (see fig. 5).

20 The multiple branch execution will require some additional hardware in the processor, since one or more partial operation is executed according to more than one part of the program simultaneously. However, in traditional methods the need to include in the program no operation commands results in a lower performance of the processor. With multiple branch execution no operation commands can be avoided and a
 25 high performance can be obtained.

Alternatively, or in combination with multiple branch execution, a procedure, herein referred to as operand forwarding, can be used. To illustrate this procedure we refer to fig. 6, in which a data processing pipeline with operation pipelining is illustrated.

5 In the pipeline in fig. 6 operations containing two steps each are performed on a context 3. For the sake of clarity of this presentation, the pipeline contains only three stages 2a, 2b, 2c, of which a final stage is not visible to a programmer of the pipeline, as explained above with reference to fig. 1.

10 In a first stage 2a an initial step of a first operation is executed in a clock cycle at a time t . The initial step includes a first instruction fetch partial operation 6a1, which is a partial operation of fetching an instruction from an instruction memory 21a, and a first operand fetch partial operation 6a2, which is a partial operation of fetching at least one operand from the context 3. In this example, the operation also contains an instruction decoding partial operation, i.e. a partial operation, not depicted in fig. 6,
15 of decoding the instruction from the instruction memory. The partial operations of the initial step of the first operation result in first initial step results, i.e. an instruction R6a1 and operand R6a2.

20 In a subsequent clock cycle at a time $t + \Delta t$, in a second stage 2b a final step of the first operation is executed. The final step includes a first execute partial operation 6c1, which is a partial operation of executing the instruction R6a1 on the operands R6a2 in a logic unit 5b, and a first write back partial operation 6c2, which is a partial operation of writing back to the context 3 a result of the execution in the logic unit 5b. (Each operation can also contain a branch partial operation, which is a partial operation, not depicted in fig. 6, for upgrading a pointer in the context, which pointer is used for fetching an instruction.)

25
30 In the same clock cycle at a time $t + \Delta t$, in the second stage, an initial step of a second operation is also executed. This step includes a second instruction fetch partial op-

eration 7a1. The initial step of the second operation also includes a second operand fetch partial operation. According to the procedure of operand forwarding, it is determined whether a position in a register in the context, from which an operand is to be fetched in the second operand fetch partial operation, is identical with a destination of the first write back partial operation 6c2. If any register position, from which an operand is to be fetched in the second operand fetch partial operation, is not identical with the destination of the first write back partial operation 6c2, the second operand fetch partial operation includes fetching 7a21 the operands from the context 3. However, if any register position, from which an operand is to be fetched in the second operand fetch partial operation, is identical with the destination of the first write back partial operation 6c2, the second operand fetch partial operation includes fetching 7a22 the object of the first write back partial operation 82a. This means fetching the result of the execution 6c1 of the instruction R6a1 on the operands R6a2 in the logic unit 5b. Thus, the result of the execution of the instruction is "stolen" before the first operation is completed.

In short, where an instruction is to use a result of a preceding instruction, and fetches a value in a register to which the preceding instruction will enter a new value, an incorrect value will be obtained. Instead the result is fetched from another location, e.g. a temporary register or an ALU-result, i.e. directly in connection to an execution in the preceding instruction.

Referring to fig. 6, in the second stage 2b, the partial operations of the initial step of the second operation result in second initial step results, i.e. an instruction R7a1 and operand R7a2. In a third stage 2c, in a subsequent clock cycle at a time $t+2\Delta t$, a final step of the second operation is executed. The final step includes a second execute partial operation 7c1 and a second write back partial operation 7c2.

CLAIMS

1. A method in a processor, in which data is processed in a pipelined manner, the data being included in a plurality of contexts, comprising a first context (3), in addition to which a plurality of operations is adapted to be executed on the contexts, characterized in that the method comprises commencing an execution on the first context of a second operation before a previously commenced execution on the first context of a first operation is completed.
2. A method in a processor, in which data is processed in a pipelined manner, the data being included in a plurality of contexts, comprising a first context (3), in addition to which a plurality of operations is adapted to be executed on the contexts, characterized in that the method comprises executing an initial operation step (6a) of a first operation on the first context (3), and subsequently commencing an execution on the first context of an initial operation step (7a) of a second operation before an execution on the first context (3) of a following operation step (6b) of the first operation is completed.
3. A method according to claim 2, whereby each context passes a plurality of consecutive stages (2a-2f), whereby
 - the initial operation step (6a) of the first operation is executed on the first context (3) at a first stage (2a),
 - the following operation step (6b) of the first operation is executed on the first context (3) at a second stage (2b), and
 - the initial operation step (7a) of the second operation is executed on the first context at the second stage (2b).
4. A method according to claim 3, comprising commencing at the first stage (2a) an execution of the initial operation step (6a) of the first operation on a second

context before the execution on the first context (3) of the following operation step (6b) of the first operation is completed.

- 5 5. A method according to any of the claims 3-4, comprising receiving at the second stage a result (R6a) of an execution of the initial operation step (6a) of the first operation.
- 10 6. A method according to any of the claims 2-5, whereby at least one of the operation steps of the second operation comprises at least two alternative execution paths, and at least two of the alternative execution paths of the operation step are executed.
- 15 7. A method according to claim 6, further comprising:
 - obtaining results (R7b1, R7b2) of at least two of the executions of the alternative execution paths, and
 - determining, based on a result (R6) of an execution of an operation step of an operation initiated before the initiation of the second operation, which one of the results (R7b1, R7b2), of the executions of the alternative execution paths, an execution of an operation step of the second operation, following said operation step comprising at least two alternative execution paths, is to be based on.
- 20 8. A method according to any of the claims 3-7, whereby the processor is arranged so that the following operation step (6b) of the first operation is presented to a programmer as being executed at the first stage (2a).
- 25 9. A method according to any of the claims 1-8, wherein the first operation comprises a partial operation of executing (6c1) an instruction and a partial operation of writing (6c2) a result of the said instruction execution into a destination in a register, and the second operation comprises the partial operation of fetching (7a21, 7a22) an operand, the method comprising
- 30

- (a) determining if a position in the register, from which the operand is to be fetched (7a21, 7a22) in the second operation, is identical with the destination of the partial operation, of the first operation, of writing (6c2) a result,
- (b) if the result of the determination in step (a) is negative, fetching (7a21) the operand from the register, and
- (c) if the result of the determination in step (a) is positive, fetching (7a22) the result of the said instruction execution.

5

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200

ABSTRACT

A method in a processor is presented, in which data is processed in a pipelined manner, the data being included in a plurality of contexts, comprising a first (3), in addition to which a plurality of operations is adapted to be executed on the contexts. The method comprises executing an initial operation step (6a) of a first operation on the first context (3), and subsequently commencing an execution of an initial operation step (7a) of a second operation on the first context before an execution on the first context (3) of a following operation step (6b) of the first operation is completed.

Fig. 3

0000198-9

44 (a)

1/6

PRV030188

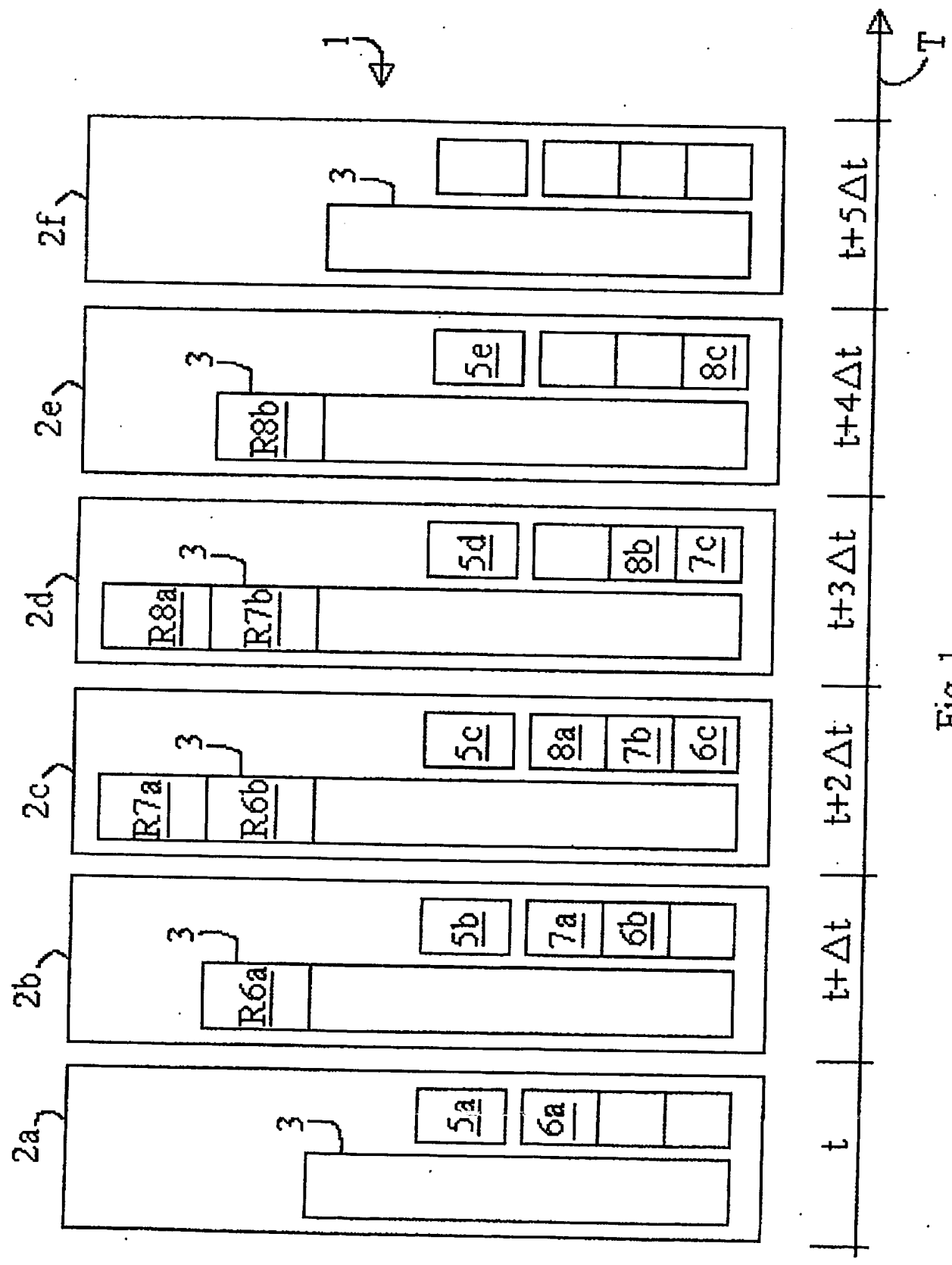


Fig. 1

2 / 6

PRU03.1.08

Instruction step	Stage at which instruction step is performed	Stage presented to a programmer as the stage at which the instruction step is performed
6a	2a	2a
6b	2b	2a
6c	2c	2a
7a	2b	2b
7b	2c	2b
7c	2d	2b
8a	2c	2c
8b	2d	2c
8c	2e	2c

Fig. 2

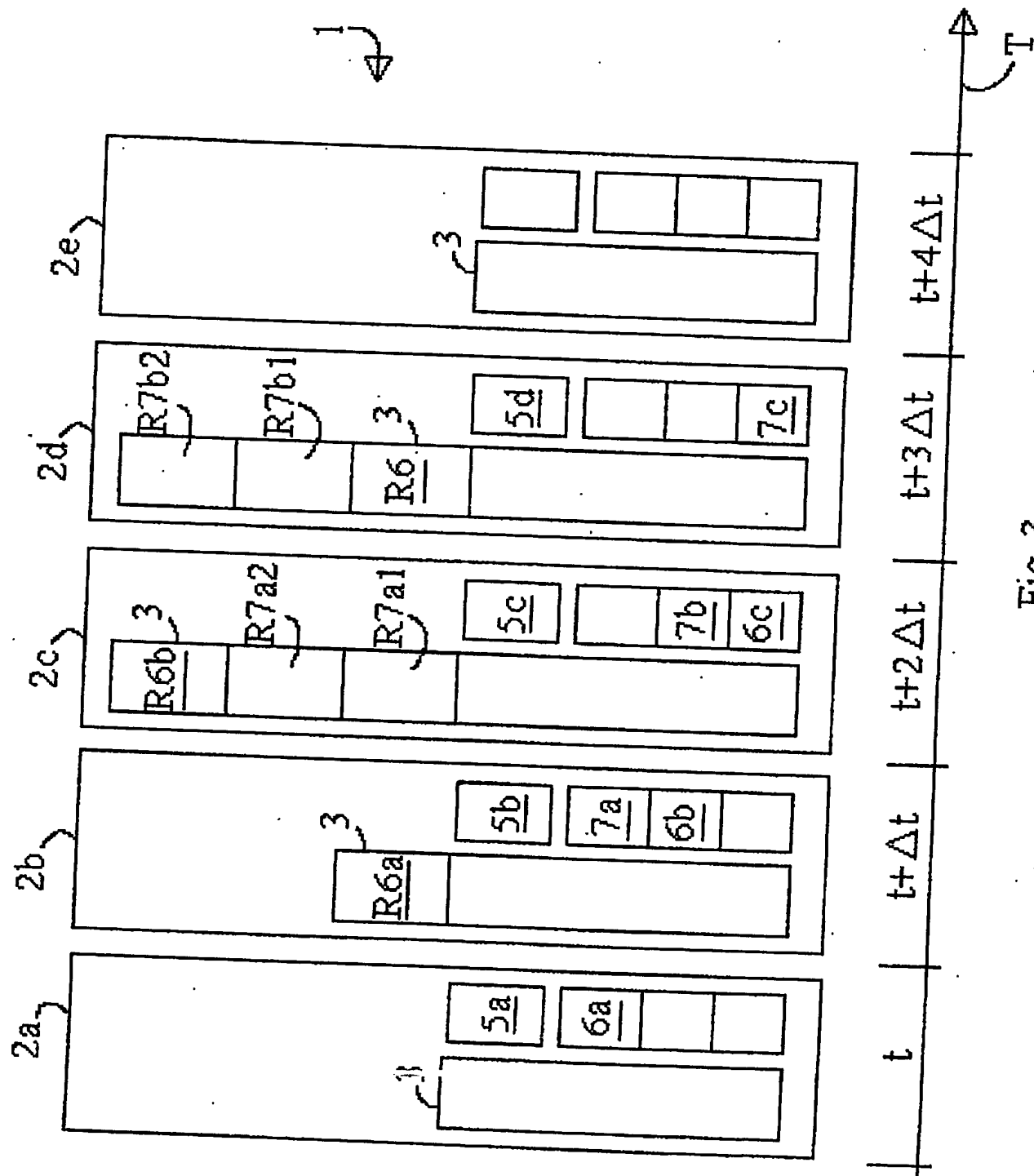


Fig. 3

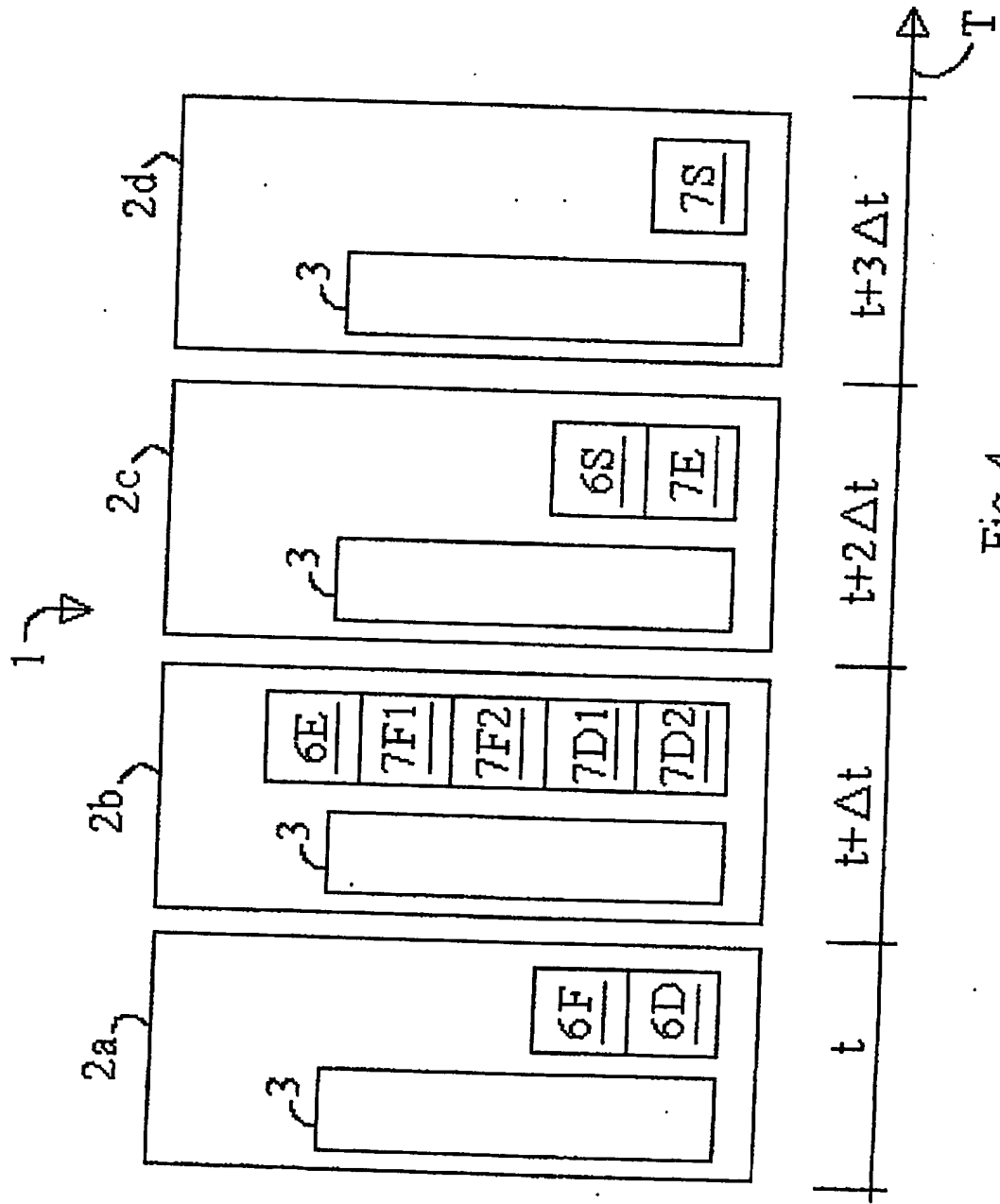


Fig. 4

5/6

PNV 10110

1	Fetch	6F
2	Decode	6D
3	If x=0 then goto L	6E
4	Fetch	7F1
5	Decode	7D1
6	Store	6S
7	Execute	7E
8	Store	7S
:		
:		
:		
:		
:		
L		
L+1	Fetch	7F2
L+2	Decode	7D2
L+3	Store	6S
L+4	Execute	7E
L+5	Store	7S

Fig. 5

000010000

000010000

000010000

000010000

6/6

000010000

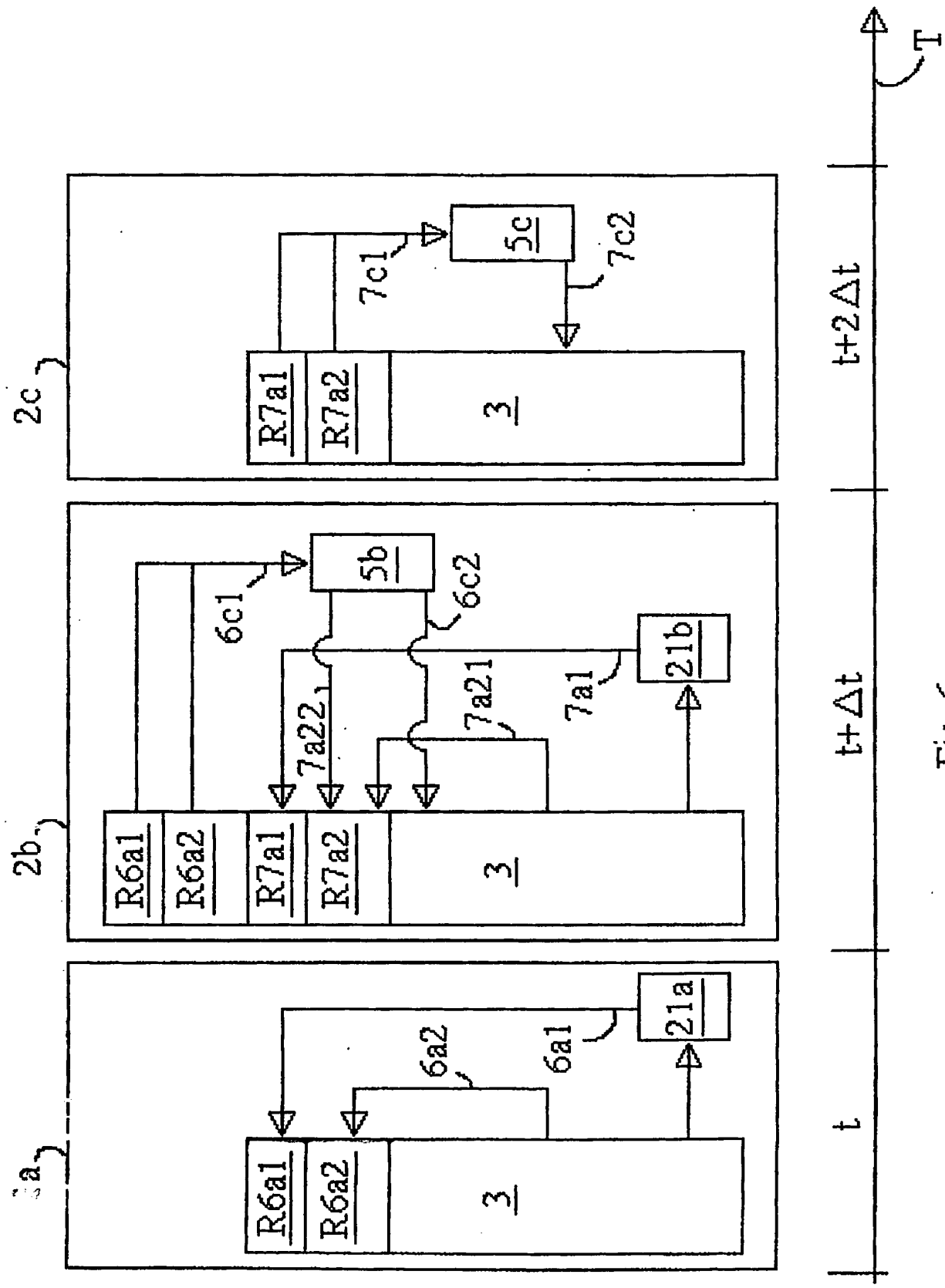


Fig. 6